Setting up NGINX for PHP

From Development Server to Production-Ready Web Server

What We're Building

Current Setup (Development)

```
php -S localhost:8000
```

- ❌ Single-threaded
- ❌ Development only
- ❌ No static file optimization
- ❌ Limited concurrent users

New Setup (Production-Ready)

```
NGINX + PHP-FPM (PHP FastCGI Process Manager)
```

- ✅ Multi-threaded
- ✅ Production ready
- ✅ Optimized static file serving
- ✅ Handles thousands of users

Understanding the Architecture

PHP Built-in Server

```
Browser → PHP -S → PHP Script
```

Single process handles everything

NGINX + PHP-FPM

```
Browser → NGINX → PHP-FPM → PHP Script
          ↓
    Static Files (direct)
```

NGINX handles static files, PHP-FPM processes PHP

Step 1: Install PHP-FPM

Windows

PHP-FPM is included with PHP 7.4+ on Windows

```
# Check if available
php --version
php-cgi --version
```

not compatible error

In the case you have an error using `php-cgi.exe`.

c:\php>.\php-cgi.exe -b 127.0.0.1:9000 -c C:\php\php.ini
PHP Warning: 'C:\WINDOWS\SYSTEM32\VCRUNTIME140.dll' 14.34 is not compatible with this PHP build linked with 14.44 in Unknown on line 0

Download and install the latest Microsoft Visual C++ Redistributable for Visual Studio 2019 or later (such as 2022):

1. Go to Microsoft's official Visual Studio download page: https://visualstudio.microsoft.com/downloads/
2. Scroll down to the "Other Tools, Frameworks, and Redistributables" section.
3. Select the appropriate version for your system (x64 for 64-bit Windows or x86 for 32-bit Windows).
4. Download and run the installer (e.g., vc_redist.x64.exe).

macOS

```
# PHP-FPM comes with Homebrew PHP
brew install php

# Or if you need to reinstall
brew reinstall php
```

```
> brew list php | grep php-fpm
/opt/homebrew/Cellar/php/8.4.11/sbin/php-fpm
/opt/homebrew/Cellar/php/8.4.11/.bottle/etc/php/8.4/php-fpm.d/www.conf
/opt/homebrew/Cellar/php/8.4.11/.bottle/etc/php/8.4/php-fpm.conf
/opt/homebrew/Cellar/php/8.4.11/.bottle/var/log/php-fpm.log
/opt/homebrew/Cellar/php/8.4.11/share/man/man8/php-fpm.8
```

Linux (Ubuntu/Debian)

```
sudo apt update
sudo apt install php-fpm php-mysql
```

Step 2: Configure PHP-FPM

Start PHP-FPM Service

macOS:

```
brew services start php
# or manually: php-fpm
```

`brew services` command shows the `brew services` that are running on your system.

```
smcho@m4 www> brew services
Name  Status  User  File
mysql started smcho ~/Library/LaunchAgents/homebrew.mxcl.mysql.plist
nginx started smcho ~/Library/LaunchAgents/homebrew.mxcl.nginx.plist
php   started smcho ~/Library/LaunchAgents/homebrew.mxcl.php.plist
```

Linux:

```
sudo systemctl start php-fpm        # CentOS/RHEL
sudo systemctl start php8.1-fpm     # Ubuntu (adjust version)
sudo systemctl enable php8.1-fpm    # Auto-start on boot
```

Windows:

Assuming that PHP is installed in c:\PHP.

1. Run php-cgi.exe.

```
cd C:\php
.\php-cgi.exe -b 127.0.0.1:9000 -c C:\php\php.ini
```

2. PHP-FPM runs automatically with proper NGINX configuration

Health Check

```
PS C:\nginx> netstat -ano | findstr :9000
  TCP    127.0.0.1:9000         0.0.0.0:0              LISTENING       93724
  TCP    127.0.0.1:9000         127.0.0.1:50074        TIME_WAIT       0

PS C:\nginx> Get-CimInstance Win32_Process -Filter "ProcessId=93724" | Select-Object ProcessId, Name, CommandLine

ProcessId Name        CommandLine
--------- ----        -----------
   93724 php-cgi.exe
```

Step 3: Basic NGINX Configuration for PHP

Find Your NGINX Configuration File

- **Windows**: `C:\tools\nginx\conf\nginx.conf` (choco) or `C:\nginx\conf\nginx.conf` (manual)
- **macOS**: `/usr/local/etc/nginx/nginx.conf` (/opt/homebrew/, instead of /usr/local/, for Apple Silicon Mac)
- **Linux**: `/etc/nginx/nginx.conf`

Create a Basic Configuration

Replace the `server` block in `nginx.conf`:

You can copy and paste the code block from module1/code/5_Webserver_using_NGINX).

```
server {
    listen 80;
    server_name localhost;
    root /var/www/html;  # Adjust path for your system
    index index.php index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        fastcgi_pass 127.0.0.1:9000;  # PHP-FPM address
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

Server Block Overview

Make sure you have the correct **root** for your system!

```
server {
    listen 8080;
    server_name localhost;
    root /usr/local/var/www;
    index index.php index.html;
}
```

- **listen 8080**: HTTP port (use 8080 on macOS instead of 80)
- **server_name localhost**: Domain (change to your own domain if needed)
- **root**: Folder to serve files (adjust path; use `/` not `\` on Windows)
- **index**: Default file(s) to load in a directory

Static File Handling

```
location / {
    try_files $uri $uri/ =404;
}
```

- Tries to serve the exact file or folder
- If not found, returns 404
- Ensures clean URL support (e.g., /about)

PHP File Handling

```
location ~ \.php$ {
    fastcgi_pass 127.0.0.1:9000;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    include fastcgi_params;
}
```

- Matches .php files via regex
- Forwards request to PHP-FPM at 127.0.0.1:9000
- Passes script path and required FastCGI parameters

How It Works

1. Browser requests http://localhost:8080/index.php
2. NGINX matches .php and forwards to PHP-FPM
3. PHP-FPM executes index.php and returns output
4. NGINX sends a response back to the browser

✅ Efficient, production-ready setup for PHP apps

Step 4: Restart nginx

- **Mac**: `nginx -s reload`
- **Linux (Ubuntu)**: `sudo systemctl restart nginx`

For windows:

- `nginx -s reload` for the simplest (two terminals) case.
- For other cases, stop and start again.

Step 5: Create a Test PHP File

Create `info.php` in your web root

```php
<?php
phpinfo();
?>
```

Create `test.php` for API testing

```php
<?php
header('Content-Type: application/json');

$response = [
    'message' => 'NGINX + PHP-FPM is working!',
    'server' => $_SERVER['SERVER_SOFTWARE'] ?? 'Unknown',
    'php_version' => PHP_VERSION,
    'timestamp' => date('Y-m-d H:i:s'),
    'method' => $_SERVER['REQUEST_METHOD'],
    'uri' => $_SERVER['REQUEST_URI']
];

echo json_encode($response, JSON_PRETTY_PRINT);
?>
```

- PHP automatically generates this as a response.

```
HTTP/1.1 200 OK
Server: nginx/1.24.0
Content-Type: application/json
Content-Length: 226
Connection: keep-alive

{
    "message": "NGINX + PHP-FPM is working!",
    "server": "nginx/1.24.0",
    "php_version": "8.2.12",
    "timestamp": "2025-08-02 18:01:30",
    "method": "GET",
    "uri": "/test.php"
}
```

Step 6: Start Services and Test

1. Start PHP-FPM (if not already running)

macOS:

```
brew services start php
```

Linux:

```
sudo systemctl start php8.1-fpm  # Ubuntu
sudo systemctl start php-fpm     # CentOS
```

Windows

Windows was explained in earlier pages.

2. Reload NGINX Configuration

macOS & Linux:

```
nginx -s reload
```

Windows was explained in earlier pages.

3. Test Your Setup

- Visit: **http://localhost/info.php** (or **:8080** on macOS)
- Visit: **http://localhost/test.php**

You should see the PHP info and the JSON response!